15-744 Computer Networks (Fall 2010)
Homework 4

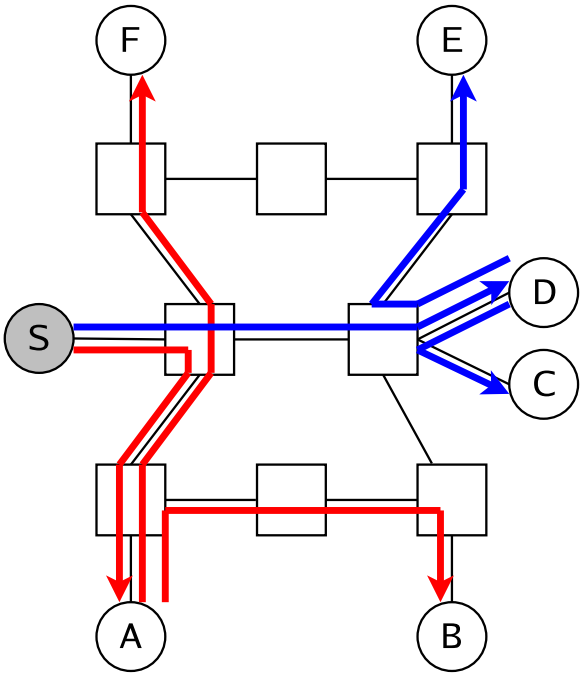
No need to hand in

Name:
Andrew ID:

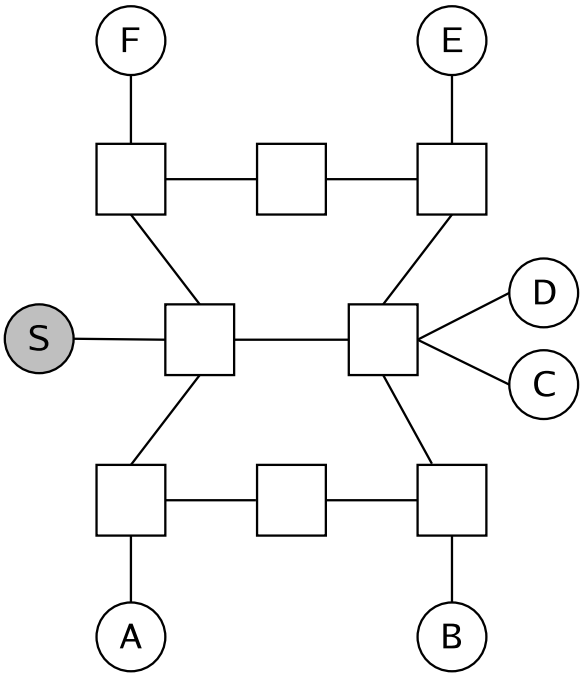
November 16, 2010

A Multicast

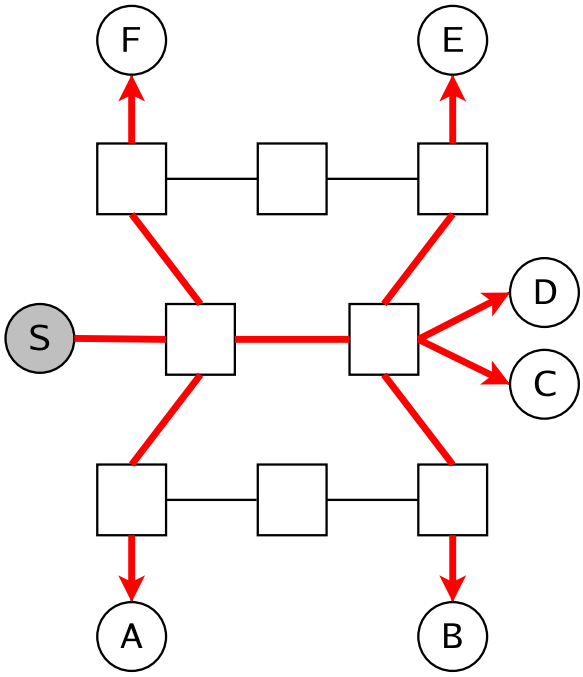
1. Consider the following topology. The squares are IP routers and the circles are end hosts which are all part of a single multicast group. S sends data to this group. Each physical link (the thin lines) has 1 unit of delay. Assume a source-based tree IP multicast protocol like MOSPF. The thick lines show an overlay multicast tree constructed on top of this topology.



(a) In the figure below, mark the edges that would be used in the IP multicast tree used when S sends data. (4 points)



Solution:



(b) The *stress* on a link is the number of duplicate packets that cross the link (in either direction). For example, the stress on S's access link is 2. What is the worst cast stress in this overlay (i.e., the stress of the most stressed link)? (3 points)

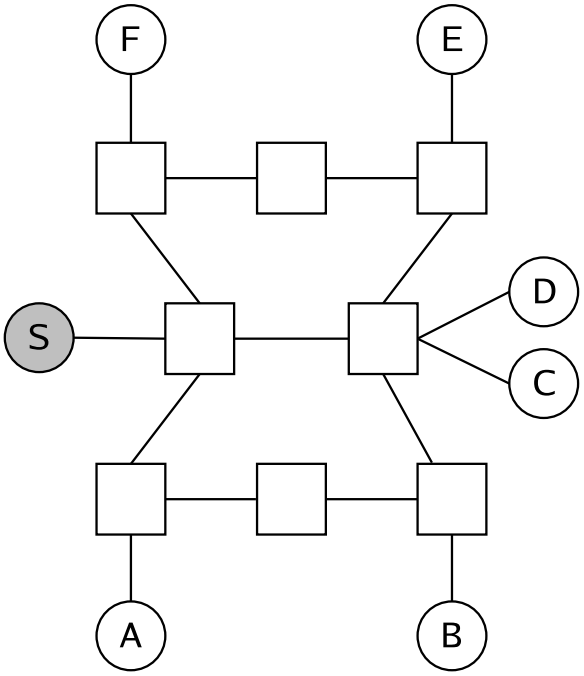
Solution: 3

(c) The *stretch* of an overlay path between two nodes is the ratio of the overlay RTT between the nodes and the actual topology RTT between the nodes. For example, the stretch from S to C is 5/3.

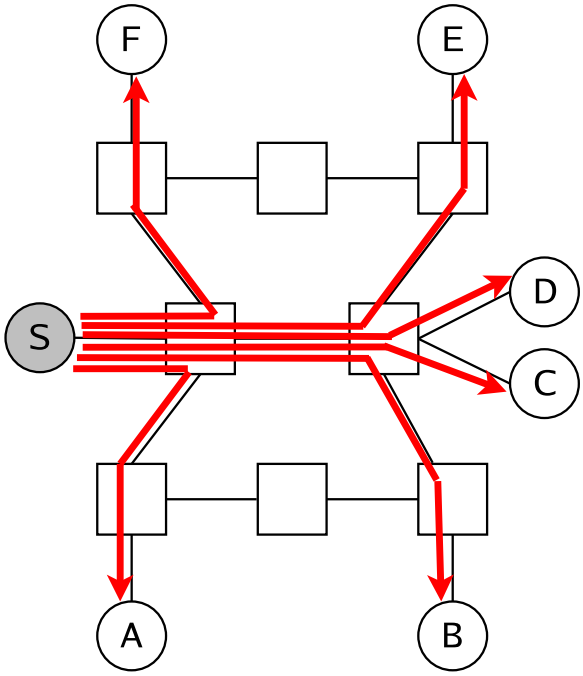
What is the stretch between S and B in this overlay? (3 points)

Solution: 7/4

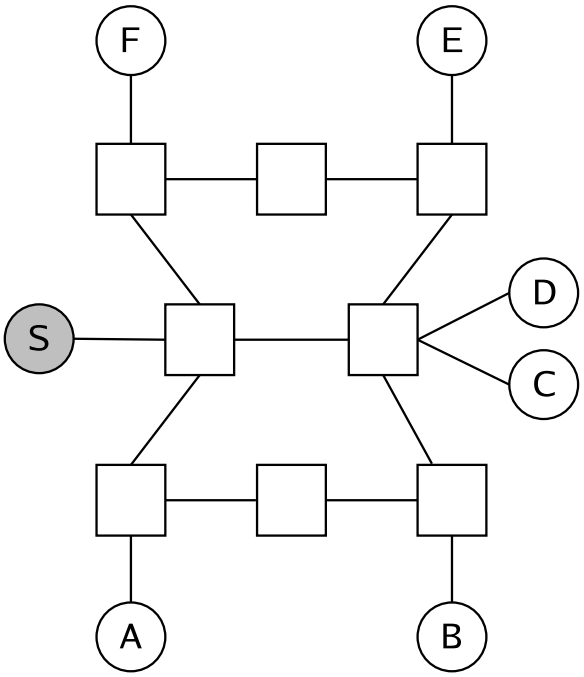
(d) In the figure below, draw an overlay multicast tree that minimizes worst case stress. (i.e., a tree that minimizes the maximum stress on any physical link) (6 points)



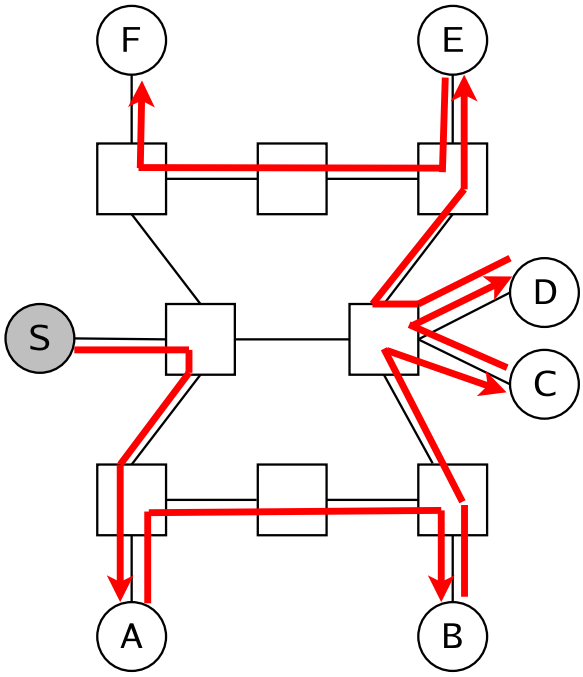
Solution:



(e) In the figure below, draw an overlay multicast tree with minimum total stretch. (i.e., a tree that minimizes the sum of stretch between the source and each end host) (6 points)



Solution:



(f) Consider a latency sensitive application. Under what conditions would you use the min-stress tree? Under what conditions would you use the min-stretch tree? Explain. (4 points)

Solution: `min-stress`: When link capacity is small compared to the application's data rate. In this case delay is mostly going to be due to queueing delay in the middle of the multicast tree, which increases with stress.

`min-stretch`: When link capacity is high compared to the application's data rate (e.g., if it is

N times the data rate, where N is the number of members). In this case, propagation delay will be the dominant factor in end-to-end latency and reducing stretch reduces propagation delay.

B DDoS

2. Suppose Alice is transfering 90,000,000 bits of data to Bob, including protocol overhead. Bob's 10 Mbit per second access link is the bottleneck and Bob's upstream router does per-source fair queuing.

   (a) How long does it take for Bob to finish the transfer, with no competing traffic?

   Solution: 4 points
   9×10^7 bits / 10^7 bits/sec = 9 seconds

   (b) Mallory doesn't want Bob to get this file quickly so his army of 99 bots launches a DDoS attack against Bob at the same time that Alice starts the transfer. Each bot sends bogus data at the same rate that Alice is sending the file at. Assume that they can't spoof IP addresses. How long does it take for Bob to finish the transfer when under attack?

   Solution: 5 points
   Fair-share means each source gets 10^7 bits/sec / 10^2 hosts = 10^5 bits/sec/host. Thus, the transfer takes 9×10^7 bits / 10^5 bits/sec = 900 seconds.

   (c) Now suppose the network supports capabilities. Assume that capability requests are 625 bytes (= 5000 bits) and each capability lasts for 100 seconds. The network performs ingress filtering to ensure that no source can send more than 2 capability requests per second.

   If Bob only grants capabilities to Alice, how long will it take for Bob to finish the transfer when under attack by Mallory? To simplify your calculation, you may assume that there are 100 bots (instead of 99). Assume that each bot tries to get capabilities as fast as possible and, if they get one, send data at the same rate as Alice.

   Solution: 5 points
   Alice sends an additional 5000 bits once to get a capability which will last long enough to finish the transfer. This takes a negligable amount of time. Each bot can send 2 capabilities (10000 bits) each second to Bob, so in aggregate they take up $100 \times 10^4 = 10^6$ bits/sec of Bob's bandwidth. Hence, the transfer takes 9×10^7 bits / $(10^7 - 10^6)$ bits/sec = 10 seconds.

C Worm Potpourri

Mallory launches his first random scanning worm, which exploits a buffer overflow in the popular Zkype VoIP program. The Zkype protocol is encrypted (i.e., so all packet payloads are indistinguishable from uniformly random bit strings) and runs on one version of Windows.

Mallory's worm looks like the following:

```
int main() {
   srand(42);    // random seed

   infect();     // infect the current host
```

```
    while (1) {
       int ipaddr = rand(); // pick a random IP address (32-bit number)
       scan( ipaddr );      // scan it and infect it if possible
       sleep( 1 );          // do this once per second
    }
}
```

3. How long will it take (on expectation) for Mallory's worm to scan all 2^{32} addresses? (You may assume the random number generator `rand()` samples without replacement if that makes your calculation easier.)

> **Solution:** Since each node uses the same random seed, they will all scan the same hosts. Hence, the scanning time is simply the time for one node to scan all the hosts, which is a coupon collector problem: about $N \log N = 2^{32} \cdot 32$ seconds.
>
> If you assume random sampling without replacement, then this is just 2^{32} seconds, since each node just iterates through the same permutation of all hosts.

4. Change one line of Mallory's code so that all hosts are scanned an order of magnitude faster.

> **Solution:** Use `srand( time()p̂id() )` instead of `srand(42)` or the like so that each host has a different random seed.

5. List 2 additional ways that Mallory could make his worm spread faster without increasing each host's scanning rate.

> **Solution:** Localized scanning, hit-list scanning, permutation scanning.

6. Alice deploys a signature based detection scheme like Early Bird at the top 40 ISPs to detect worms based on observed packet payloads. Will Alice be able to detect and block Mallory's worm? Explain.

> **Solution:** No. Since the packet payloads are encrypted, no two packets carrying the worm will be any more similar than any two legitimate packets.

D TCP Security

7. Recall that the TCP connection setup consists of a three-way handshake.

   If a client A was trying to connect to some server S the three-way handshake would look something like this:

   $A \rightarrow B : SYN(ISN_A)$

   $B \rightarrow A : ACK(ISN_A), SYN(ISN_S)$

   $A \rightarrow B : ACK(ISN_S)$

(a) A common DoS attack is flooding a server with TCP SYN packets. Briefly explain what makes the flooding of TCP SYN packets more effective than flooding other types of packets (such as "ping" packets)?

Solution: TCP SYN packets requires a server to keep state of a client connecting to it so that the server can later verify the client's next message. Thus, TCP SYN packets make a server use up memory resources. This is different from a "ping" packet which only requires a server to respond with another packet but does not need to keep any kind of state.

(b) A particular server always returns the same initial value ISN_S for every client trying to setup a TCP connection with it. State the name of the attack that is possible against such a server.

Solution: Session hijack